LArSoft - Bug #23279

LArSim revision/commit d96375d3 breaks DUNE analysis.

09/16/2019 07:57 PM - Jason Stock

Status:	Closed	**Start date:**	09/16/2019
Priority:	High	**Due date:**	
Assignee:		**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:		**Spent time:**	1.00 hour
Occurs In:		**Co-Assignees:**	
Experiment:	-		

Description

As stated, Revision d96375d3 is reported to break analysis in use by DUNE as we have simulations that do not contain `sim::GeneratedParticleInfo`. Is this a feature we are lagging behind with, or is this perhaps an experiment specific inclusion? This was first identified by David Rivera. I would like to revert this if the inclusion of `sim::GeneratedParticleInfo` isn't universally compatible.

History

#1 - 09/16/2019 10:01 PM - Gray Putnam

Hello -- I am the person who committed this change to larsim. This change was not experiment specific -- all of SBND, ICARUS, and uBooNE generate MC with the sim::GeneratedParticleInfo object present. This commit was made to fix a bug which caused the ParticleInventoryService to fail when used inside of gallery: the gallery::Event class does not allow for the “data” object in associations to be omitted when retrieving a set of Assns. Specifying this class in the getValidHandle call thus allows gallery to use the BackTracker service without crashing.

Also I am not sure if this is the correct way to discuss issues in redmine -- please let me know if there is a better way to post this comment.

#2 - 09/16/2019 10:13 PM - Jason Stock

Thanks for commenting! We will need to figure out how to make this work for all cases then. We definitely need to make sure we support Gallery, but I wonder if there is a more general way to do so that doesn't fail when the data field doesn't exist. Perhaps we need to file a bug with Gallery?

#3 - 09/17/2019 10:22 AM - Jason Stock

drivera circulated a potential fix for this with feature/drivera_bug_23279_fix, but, it does make use of try/catch that will for DUNE result in a catch every event. I would like to avoid doing so if possible.

#4 - 09/17/2019 11:34 AM - David Rivera

I mainly wrote it with the catch on the DUNE case to test it with available data. If we agree that the default behaviour should be to not catch on DUNE but rather on gallery use cases then I can switch the order of the try statements. Could the determination of whether or not we should require GeneretedParticleInfo when getting the MCTruth, MCParticle Assns be done only once and not for each event? I don't think that we expect associations with and without GeneretedParticleInfo within the same file.

#5 - 09/17/2019 12:25 PM - Kyle Knoepfel

Look like some cleanup of ParticleInventory.tcc is in order. As it appears that the sim::GeneratedParticleInfo data in the Assns is not actually *used* in that file, the best approach is to just remove the sim::GeneratedParticleInfo template argument during the product retrieval call:

```
- evt.template getValidHandle<art::Assns<simb::MCParticle,simb::MCTruth, sim::GeneratedParticleInfo
>>(fG4ModuleLabel)
+ evt.template getValidHandle<art::Assns<simb::MCParticle,simb::MCTruth>>(fG4ModuleLabel)
```

Even though the actual Assns on disk may have the sim::GeneratedParticleInfo metadata associated with it, *art* allows you to retrieve the Assns without the metadata.

Bottomline: Remove the sim::GeneratedParticleInfo from the getValidHandle call and get rid of all of the try/catch blocks and switches based on whether Assns<A, B> or Assns<A, B, D> has been written to disk.

#6 - 09/17/2019 12:41 PM - Kyle Knoepfel

- Status changed from New to Feedback

Try larsim branch feature/knoepfel_23279_fix.

#7 - 09/17/2019 12:48 PM - Jason Stock

Thanks for chiming in Kyle. So doing this would revert exactly back to the behaviour pre d96375d3, which is what Gray informed us was breaking Gallery, and why he originally changed ParticleInventory.tcc. (Diff of your branch with develop before Gray's commit included below for discussion)

```
     void ParticleInventory::PrepMCTruthListAndTrackIdToMCTruthIndex(const Evt& evt ) const{
       if( this->TrackIdToMCTruthReady() && this->MCTruthListReady( ) ){ return;}
       this->PrepParticleList( evt); //Make sure we have built the particle list for this event
-
      //const auto& mcpmctAssnsIn = *( evt.template getValidHandle<art::Assns<simb::MCParticle,simb::MCTruth>>
(fG4ModuleLabel));
-
      const auto& mcpmctAssnsHandle =  evt.template getValidHandle<art::Assns<simb::MCParticle,simb::MCTruth>>
(fG4ModuleLabel);
-      const auto& mcpmctAssnsIn = *mcpmctAssnsHandle;
+
      const auto& mcpmctAssnsIn = *( evt.template getValidHandle<art::Assns<simb::MCParticle,simb::MCTruth>>(f
G4ModuleLabel));
       for( const auto& mcpmctAssnIn : mcpmctAssnsIn){    //Assns are themselves a container. Loop over entrie
s.
         const art::Ptr<simb::MCParticle>& part=mcpmctAssnIn.first;
         const art::Ptr<simb::MCTruth>&    mct =mcpmctAssnIn.second;
@@ -94,6 +98,10 @@ namespace cheat{
```

I agree that we should absolutely avoid the try/catch. If this call doesn't work without the `sim::GeneratedParticleInfo` in gallery then a bug report should be filed and it should be addressed upstream, not in the ParticleInventory.

Gray, would you be willing to test this in Gallery, and post the exact failure message that you get?

#8 - 09/17/2019 12:56 PM - Kyle Knoepfel

Ah, yes you will have troubles with gallery. In that case, you should use getByLabel:

```
Handle<Assns<A,B,D>> h1;
if (e.getByLabel(h1, label)) { // Product fetch successful
  auto const& abd = *h1;
}

Handle<Assns<A, B>> h2;
if (e.getByLabel(h2, label)) {
  auto const& ab = *h2;
}
```

Better yet would be to refactor the functionality so that it doesn't depend on art or gallery at all, and you move all product fetching into framework- or gallery-aware code.

#9 - 09/17/2019 01:07 PM - Jason Stock

As you and I have already discussed before, re-factoring the backtracker suite is definitely on the table if there is someone to do it, but I still think we want to wait until the consumer/consumes/provides model is completely implemented to do so as we already want to re-factor it in such a way as to make it thread safe, and to demote it from service to toolkit.

I hadn't used getByLabel for Assns before, but it looks like a winning solution for now.

#10 - 09/17/2019 02:35 PM - David Rivera

Giving it a try with getByLabel as suggested by Kyle Knoepfel. The feature branch is: feature/drivera_bug_23279_suggested_knoepfel_fix

I threw in a return at the end of each attempt so as to avoid a second getByLabel attempt if the first one is successful.

Also, the getByLabel function takes the label as its first argument and the handle as the second argument. So something like:

```
Handle<Assns<A,B,D>> h1;
-if (e.getByLabel(h1, label)) { // Product fetch successful
+if (e.getByLabel(label, h1)) { // Product fetch successful
  auto const& abd = *h1;
}

Handle<Assns<A, B>> h2;
-if (e.getByLabel(h2, label)) {
+if (e.getByLabel(label, h2)) {
  auto const& ab = *h2;
}
```

I can post the full diff between the current code and my feature branch here if that's easier than pulling my feature branch. I tested the new implementation with my analyzer and this seems to work. It should be tested within gallery as well, but I can't do that easily.

#11 - 09/18/2019 03:27 PM - Lynn Garren

What is the status of this issue? Is everyone happy with feature/drivera_bug_23279_suggested_knoepfel_fix?

I ran the CI tests with feature/drivera_bug_23279_suggested_knoepfel_fix. Those all seem to be fine, but I don't think they test the actual problem.

#12 - 09/18/2019 03:34 PM - David Rivera

I have tested this within Larsoft with data that doesn't contain the GeneratedParticleInfo association. Someone needs to test the ParticleInventory service within gallery. It would be nice if Gray could give it a shot since he needed the patch.

#13 - 09/19/2019 02:29 PM - David Rivera

I did a quick test of the particle inventory service in gallery by making a call to the PrepMCTruthList() function which calls the function in question (PrepMCTruthListAndTrackIdToMCTruthIndex); however, I got errors.

```
/dune/app/users/drivera/latest/localProducts_larsoft_v08_30_02_e17_prof/larsim/v08_12_04/include/larsim/MCChea
ter/ParticleInventory.tcc:92:9: error: no matching function for call to ‘gallery::Event::getByLabel(const art:
:InputTag&, art:
:Handle<art::Assns<simb::MCParticle, simb::MCTruth, sim::GeneratedParticleInfo> >&) const’
         if (evt.getByLabel(fG4ModuleLabel, mcpmctAssnsHandle_g)) { // Product fetch successfull
         ^~
In file included from BackTracker.cc:24:0:
/cvmfs/larsoft.opensciencegrid.org/products/gallery/v1_12_04/include/gallery/Event.h:65:10: note: candidate: t
emplate<class PROD> bool gallery::Event::getByLabel(const art::InputTag&, gallery::Handle<PROD>&) const
     bool getByLabel(art::InputTag const&, Handle<PROD>& result) const;
```

gallery does not like to be handed an art::Handle, it requires a gallery::Handle. We will have to find a way to make the determination of what type of handle to give the getByLabel function if we want to use Kyle's suggestion. My tests within larsoft worked because I declared the handles as art::Handles and included the necessary header file in ParticleInventory.tcc which is already suboptimal given that it's a template file intended to allow for art::Event and gallery::Events to be used. Not sure how to proceed from here, suggestions are welcome.

#14 - 09/19/2019 03:51 PM - Lynn Garren

I am getting confused. No software in larsim should be used when working with gallery. Only software in larcoreobj, larcorealg, lardataobj, or lardataalg is meant to work with gallery.

#15 - 09/19/2019 04:07 PM - Jason Stock

The Backtracking suite is entirely in larsim/MCCheater, including the art independent service providers (which should probably live in alg). Backtracker, PhotonBacktracker, and ParticleInventory are supposed to be entirely art independent.

#16 - 09/19/2019 04:22 PM - Lynn Garren

Thanks for the clarification. Should this branch go into to the larsoft release with Jason's latest fixes? Or do you want to wait? We're essentially out of time. I have to tag very soon.

Lynn

#17 - 09/19/2019 04:26 PM - Jason Stock

Hi Lynn,

This will probably have to miss this weeks release. David and I are meeting right now to try and come up with a best solution.

#18 - 09/19/2019 04:29 PM - Jason Stock

Kyle, is there a reason why we can use `*e.getValidHandle*` in art to get something from the principle of type `*Assns<A,B>*` when the principle in fact contains `*Assns<A,B,D>*`, but not in gallery? There is no need for the *simb::GeneratedParticleInfo* in the ParticleInventory, and the only reson for adding it to this code was to make it work for Gallery. It seems like the correct thing is to revert to supporting the art case, and file an issue with gallery for "_`e.getValidHandle`_ fails for `*Assns<A,B,D>*` when only `*Assns<A,B,void>*` requested".

#19 - 09/19/2019 07:35 PM - Kyle Knoepfel

David, you can solve the gallery::Handle vs. art::Handle problem by using the nested HandleT<T> class template contained by both gallery and art events:

```
template <typename Event>
void processEvent(Event& e)
{
  typename Event::HandleT<MyType> h;  // Should work for either art or gallery events
  if (e.getByLabel(some_label, h)) {
    ...
  }
}
```

Jason, the relaxed Assns-lookup rules in *art* were implemented at a time when *gallery* was still under development. The *gallery* library is a near clone of the facility CMS has implemented for its own files. So although the product-lookup behavior is intended to be the same, the implementation in *gallery* is very different from that in *art*. I would love to use the same product-lookup functionality in both *art* and *gallery*, but that is a good deal of work. Nonetheless, I encourage you to make a gallery feature request.

#20 - 09/19/2019 08:00 PM - Jason Stock

Thank you for your answers Kyle. That is extremely helpful. David and I will try to get this working as soon as possible, and I will definitely file the feature request with Gallery.

#21 - 09/19/2019 09:05 PM - David Rivera

Thank you, Kyle. I've incorporated the templated Handle and have tested using the ParticleInventory on files that both contain and don't contain the GeneratedParticleInfo within larsoft and within gallery. As suggested by Jason, the order is: try with the relaxed method, else try with the more strict method and if both of these fail, throw a cet::exception.

I've pushed the changes to: **feature/drivera_bug_23279_suggested_kyle_fix** for others to take a final look.

#22 - 09/23/2019 10:34 AM - Kyle Knoepfel

Is this branch now ready for this week's LArSoft release?

#23 - 09/23/2019 11:26 AM - David Rivera

Hi Kyle, according to my tests with larsoft and gallery, yes.

the branch is: **feature/drivera_bug_23279_suggested_kyle_fix**

#24 - 09/23/2019 11:45 AM - Lynn Garren

Super! We are scheduling this for inclusion in this weeks' release.

#25 - 09/26/2019 11:52 AM - Lynn Garren

- % Done changed from 0 to 100

- Status changed from Feedback to Resolved

These changes are in larsim v08_13_01 (larsoft v08_31_01).

#26 - 09/30/2019 10:50 AM - Kyle Knoepfel

- Subject changed from LArSim revission d96375d3 breaks DUNE analysis. to LArSim revision/commit d96375d3 breaks DUNE analysis.

#27 - 10/14/2019 10:40 AM - Lynn Garren

- Status changed from Resolved to Closed